

8-11-00

A

## UTILITY PATENT APPLICATION TRANSMITTAL

(NO FEE)

(New Nonprovisional Applications Under 37 CFR § 1.53(b))

Attorney Docket No.

50325-0114

## TO THE COMMISSIONER FOR PATENTS:

Transmitted herewith is the patent application of ( ) application identifier or (X) first named inventor, David N. Still, et al., entitled SELECTIVELY PASSING NETWORK ADDRESSES THROUGH A SERVER, for a(n):

(X) Original Patent Application.

( ) Continuing Application (prior application not abandoned):

( ) Continuation ( ) Divisional ( ) Continuation-in-part (CIP)  
of prior application No: \_\_\_\_\_ Filed on: \_\_\_\_\_

( ) A statement claiming priority under 35 USC § 120 has been added to the specification.

## Enclosed are:

(X) Specification 27 Total Pages; (X) Drawing(s); 8 Total Sheets; (X) Cover Sheet 1 Page

(X) Oath or Declaration: 3 Pages

( ) A Newly Executed Combined Declaration and Power of Attorney:

( ) Signed. (X) Unsigned. ( ) Partially Signed.

( ) A Copy from a Prior Application for Continuation/Divisional (37 CFR § 1.63(d)).

( ) Incorporation by Reference. The entire disclosure of the prior application, from which a copy of the oath or declaration is supplied, is considered as being part of the disclosure of the accompanying application and is hereby incorporated herein by reference.

( ) Signed Statement Deleting Inventor(s) Named in the Prior Application. (37 CFR § 163(d)(2)).

( ) Power of Attorney.

(X) Return Receipt Postcard.

( ) Associate Power of Attorney.

( ) A Check in the amount of \$ \_\_\_\_\_ for the Filing Fee.

( ) Preliminary Amendment.

( ) Information Disclosure Statement and Form PTO-1449.

( ) A Duplicate Copy of this Form for Processing Fee Against Deposit Account.

( ) A Certified Copy of Priority Documents (if foreign priority is claimed).

( ) Statement(s) of Status as a Small Entity.

( ) Statement(s) of Status as a Small Entity Filed in Prior Application, Status Still Proper and Desired.

( ) Other: \_\_\_\_\_

CLAIMS AS FILED				
FOR	NO. FILED	NO. EXTRA	RATE	FEE
Total Claims	25	5	\$18.00	\$ 90.00
Independent Claims	6	3	\$78.00	\$ 234.00
Multiple Dependent Claims (if applicable)				\$ 0.00
Basic Filing Fee				\$ 690.00
Total Filing Fee				\$ 1014.00

## NO FEES WILL BE PAID AT THIS TIME.

Charge \$ \_\_\_\_\_ to Deposit Account 50-1302 pursuant to 37 CFR § 1.25. At any time during the pendency of this application, please charge any fees required or credit any overpayment to this Deposit Account.

Respectfully submitted,

By: \_\_\_\_\_

Christopher J. Palermo, Reg. No. 42,056

Date: August 9, 2000

Correspondence Address:

Hickman Palermo Truong & Becker, LLP  
1600 Willow Street  
San Jose, California 95125-5106  
Telephone: (408) 414-1080  
Facsimile: (408) 414-1076

I hereby certify that this is being deposited with the U.S. Postal Service "Express Mail Post Office to Addressee" service under 37 CFR § 1.10 on the date indicated below and is addressed to:

Commissioner for Patents  
Box Patent Application  
Washington, D.C. 20231

By: \_\_\_\_\_

Typed Name: Casey Moore

Express Mail Label No.: EL624353556US

Date of Deposit: August 9, 2000

50325-0114 (1115 / 33981)

Patent

UNITED STATES PATENT APPLICATION

FOR

SELECTIVELY PASSING NETWORK ADDRESSES THROUGH A SERVER

INVENTORS:

DAVID N. STILL  
MINQI DENG  
THOMAS A. RODEN

PREPARED BY:

HICKMAN PALERMO TRUONG & BECKER LLP  
1600 WILLOW STREET  
SAN JOSE, CA 95125-5106  
(408) 414-1080

EXPRESS MAIL CERTIFICATE OF MAILING

"Express Mail" mailing label number EL624353556US

Date of Deposit August 9, 2000

I hereby certify that this paper or fee is being deposited with the United States Postal Service "Express Mail Post Office to Addressee" service under 37 CFR 1.10 on the date indicated above and is addressed to the Assistant Commissioner for Patents, Washington, D.C. 20231.

Casey Moore

(Typed or printed name of person mailing paper or fee)

  
(Signature of person mailing paper or fee)

# SELECTIVELY PASSING NETWORK ADDRESSES THROUGH A SERVER

## RELATED APPLICATION

This application is a Continuation-in-part of co-pending application Ser. No.  
5 09/225,909, filed January 05, 1999, "Selectively Forced Re-direction of Network Traffic," by  
D. Still et al., and domestic priority is hereby claimed from such application.

## FIELD OF THE INVENTION

The present invention generally relates to data processing. The invention relates more  
specifically to selectively passing network addresses through a server in a networked  
10 computer system.

## BACKGROUND OF THE INVENTION

In a client/server computing environment in which clients communicate with servers  
over non-secure networks, enabling only authorized clients to receive services from the  
servers presents a variety of technical problems. In one approach, identifying information is  
15 collected from each client, and only known clients are given services from a server. In one  
implementation of this approach, a host that originates a request for services must provide its  
source IP address to the receiving server.

However, in this approach, there is a need to minimize the number of times that the  
client identifying information travels across the network, in order to prevent interception and  
20 spoofing. For example, allowing the IP address to be passed explicitly as part of a secure  
protocol message could represent a security weakness; a client could intentionally supply an  
incorrect address to gain greater access. Unfortunately, not allowing the address to be passed  
explicitly at all means that important identifying information is not available to the receiving  
server.

25 FIG. 1A is a simplified block diagram of a client/server system in which this general  
problem may arise. One or more clients 1112A, 1112B are coupled to a network 1104. Each

client 1112A, 1112B is a network end station device such as a personal computer, workstation, or the equivalent. Alternatively, each client 1112A, 1112B is a process, such as a standard Web browser. The term "originating host" is used herein to refer to any of clients 1112A, 1112B. Although only two clients are shown in FIG. 1A, in a practical system there may be any number of such clients.

Network 1104 is one or more local area networks, wide area networks, or internetworks, alone or in combination. In a preferred embodiment, network 1104 is the global, packet-switched internetwork known as the Internet.

An HTTP server 1130 is coupled to network 1104 in a position logically separate from clients 1112A, 1112B. HTTP server 1130 comprises one or more servers or software elements that can respond to client requests received in the Hypertext Transfer Protocol (HTTP). HTTP server 1130 may have one or more associated application servers that provide other services.

HTTP server 1130 stores or manages session data 1140, which comprises a plurality of records that identify clients that are authorized to receive services from the HTTP server. In an embodiment, session data 1140 is created and stored by HTTP server 1130 for the purpose of uniquely identifying clients that are authorized to access HTTP server 1130, its resources, or other associated servers and resources. Each record in session data 1140 includes a key 1141 that includes, among other data, client IP address values 1144A, 1144B associated or paired with random number values 1142A, 1142B. For purposes of illustrating an example, two (2) keys 1141 are shown in FIG. 1A, however, in a practical system there may be any number of keys or other information in session data 1140.

Each IP address value 1144A, 1144B normally represents an Internet Protocol (IP) address that is pre-assigned to and uniquely associated with one of the clients 1112A, 1112B. In conventional usage, IP addresses are uniquely associated with specific client hardware such as a particular workstation or personal computer. However, such a hardware device may

execute a plurality of instances of client applications, such as Web browsers. Accordingly, random values 1142A, 1142B may be used in order to uniquely identify more than one instance of a browser that is running on the same physical machine. Random values 1142A, 1142B are generated by HTTP server 1130 when a client having an authorized IP address connects to the HTTP server.

FIG. 1B is a simplified block diagram of a system similar to that of FIG. 1A, in which like numbered blocks represent like elements. In FIG. 1B, however, a proxy server 1108 is coupled to network 1104 for the purpose of providing services that complement, but are not available in, an application server 1106, which is also coupled to network 1104.

Proxy servers are useful for merging functionality from different servers. For example, assume that application server 1106 offers Secure Sockets Layer functions, but not servlet capabilities, and proxy server 1108 can provide servlet functions. In this case, proxy server 1108 can act as proxy for application server 1106. Proxy server 1108 receives such requests from clients 1112A, 1112B and can respond to them. However, to carry out a response, proxy server 1108 may need a service or information from application server 1106, by communication over logical path 1110. When application server 1106 receives a service request from proxy server 1108, the application server stores the IP address of the proxy server as part of a key of a record in session data 1140.

In a security scheme that requires the source IP address of the originating host, a system that includes a proxy server can result in problems. In particular, security can be compromised because the receiving server (e.g., application server 1106) always receives the IP address of the proxy server 1108 with a service request, rather than the IP address of the clients 1112A, 1112B that originate service requests.

One responsive measure is not allowing the address to be passed explicitly at all in the protocol, however, in that case important identifying information is not available to the

receiving server. If the address is passed, security is reduced because all accesses appear to originate from the proxy server.

Based on the foregoing, there is a need in this field for a way to pass originating client or host network address information through a proxy server to a receiving server, in a secure  
5 manner.

There is a specific need for a way to pass the IP address of a Web client through a proxy server to an HTTP server or application server, in a secure protocol that does not always allow the IP address to be passed.

In the client/server computing environment a need may arise to enable a client to  
10 communicate with two servers, each of which provide functions that represent a portion of a service desired by the client. In some cases, it is desirable to permit the client to communicate with only a first one of the two servers and to prevent direct contact between the client and the other of the two servers.

FIG. 1C is a block diagram of a networked computer system in which the foregoing  
15 general problem may arise. Client 102 is a computer device such as a workstation, server, router, or switch. Client 102 is coupled to network 104, which is an interconnected combination of computers, terminals, or peripherals used to provide communications between two or more points. A first server 106 and a second server 108 are coupled to network 104, logically separated from client 102. In one embodiment, client 102, network  
20 104, and servers 106, 108 communicate using TCP/IP network protocols, and using HTTP protocol messages that comprise requests and responses. Such protocols are exemplary and not required.

Client 102 may communicate an HTTP request for a service to network 104. The request includes a name or other identifier of server 106, which client 102 expects to provide  
25 the requested service. Network 104 locates server 106 and routes the request to server 106, thereby establishing a logical connection 110 from client 102 to the server. Server 106

determines that it cannot provide the function or service solicited in the request, but that server 108 can provide the function or service. Accordingly, software elements in server 106 automatically divert or "redirect" the request to server 108 over a logical connection 112, which may physically travel through the network 104. Server 108 processes the request, generates a response message, and sends the response message back to client 102 over logical connection 114, which may pass through network 104.

In this scenario, when the client and the servers use HTTP, the response message may include a document formatted using a structured markup language, such as HTML. The HTML document may contain hyperlinks or other references to resources within server 108, or other servers or network elements. As a result, client 102 may select one of the hyperlinks or references, and thereby attempt to request a service of server 108 directly along connection 114. Server 106 would not be involved in processing such a request.

This result is undesirable in several circumstances. The server 106 may have been designated as authoritative for certain kinds of transactions. The server 106 may have redirected the original request to server 108 solely because server 106 cannot directly process the request, whereas server 108 can, but server 106 may need to remain in control of the overall transaction. For example, server 106 may have redirected the original request to server 108 just to carry out a specialized or subordinate task, although server 106 remains responsible for the total transaction or for presenting a consistent interface to the client 102.

Accordingly, there is a need in this field for a mechanism that allows a client request to be redirected from a first server to a second server, while keeping the first server in control of subsequent requests by the client for services of the second server.

In particular, there is a need for a mechanism that prevents the client from directly communicating with the second server even after the first server has redirected a request of the client to the second server.

There is a specific need in Internet protocol networks, such as Intranets or the Internet in which clients and servers use TCP/IP and HTTP, to force subsequent requests resulting from HTML generated by the second server to come back to the first server for further redirection.

5



## SUMMARY OF THE INVENTION

The foregoing needs and objects, and other needs and objects that will become apparent from the following description, are achieved by the present invention, which comprises, in one aspect, a method of securely communicating a network address of a client

5 that issues service requests to a first server that proxies the service requests for a second server. A network address of the client is received. A processor determines whether a first network address of the first server is equal to a second network address of the second server. The network address of the client is sent from the first server to the second server in a secure request message only when the first network address of the first server is equal to the second

10 network address of the second server. Accordingly, a secure communications protocol is provided in which an address of a requesting client, e.g., an IP address, is passed in the protocol only among a responding server and its proxy, thereby preventing interception of the client IP address by unauthorized processes. By enforcing a policy that permits the network address of an originating host to pass from a first server to a second server only when the

15 network address of the second server meets specified criteria (e.g., it is the same network address as that of the first server), the originating host address can be passed securely through a proxy server.

## BRIEF DESCRIPTION OF THE DRAWINGS

The present invention is illustrated by way of example, and not by way of limitation, in the figures of the accompanying drawings and in which like reference numerals refer to similar elements and in which:

5           FIG. 1A is a simplified block diagram of a networked computer system;

FIG. 1B is a simplified block diagram of a networked computer system that includes a proxy server;

FIG. 1C is a simplified block diagram of another networked computer system;

10           FIG. 2 is a block diagram of a networked computer system according to an embodiment;

FIG. 3A is a flow diagram of a process of selectively forced redirection of network traffic;

FIG. 3B is a flow diagram of further steps in the process of FIG. 3A;

15           FIG. 3C is a flow diagram of a method of selectively allowing a source network address to pass through a server; and

FIG. 4 is a block diagram of a computer system with which an embodiment of the invention may be implemented.

## DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENT

A method and apparatus for selectively passing network addresses through a server is described. In the following description, for the purposes of explanation, numerous specific details are set forth in order to provide a thorough understanding of the present invention. It will be apparent, however, to one skilled in the art that the present invention may be practiced without these specific details. In other instances, well-known structures and devices are shown in block diagram form in order to avoid unnecessarily obscuring the present invention.

### -- SELECTIVELY PASSING NETWORK ADDRESSES

FIG. 3C is a flow diagram of a method of selectively allowing a source network address to pass through a server.

In block 340, a network address is received from a host that is originating a service request. For example, application server 1106 may receive a Universal Resource Locator (URL) or HTML POST form that includes a service request message, in which one parameter or field value is the IP address of the requesting host. The requesting host may be one of the clients 1112A, 1112B, a Web browser executed by one of them, etc.

In block 342, a determination is made whether the network address of a destination server specified in the request is the same as the network address of the current server. In one embodiment, the service request message is received at a proxy server that proxies requests for the application server. Computer program code or other software or hardware elements determine whether an IP address of a server specified in the request message is identical to the IP address of the proxy server. This will occur, for example, when the proxy server and application server are hosted on the same computer hardware. If the service request of the host is provided in the form of a URL, then block 342 may include carrying out ancillary steps, e.g., performing a DNS lookup of the IP address that corresponds to the hostname portion of the URL, etc.

In block 344, if the test of block 342 yields a positive result, then the network address of the originating host is passed in a service request message from the current server to the destination server. Block 344 may involve creating an HTTP request message or URL that includes the client IP address. The service request message is forwarded or otherwise  
5 communicated to the destination server in block 346.

If the test of block 342 yields a negative result, then the originating host is not allowed to send its network address to the destination server. In one embodiment, an error response message is sent back to the originating host, as shown by block 358. Block 358 may involve returning an HTTP response message with an error code, or any other suitable means  
10 for signaling from the proxy server to the originating host that an error has occurred, or that security rules prohibit forwarding or responding to the request.

Accordingly, by enforcing a policy that permits the network address of an originating host to pass from a first server to a second server only when the network address of the second server meets specified criteria (e.g., it is the same network address as that of the first  
15 server), the originating host address can be passed securely through a proxy server. As long as closed criteria are used for specifying when explicit network address passing is allowed, the address passing system can be secure.

#### -- SELECTIVELY FORCED REDIRECTION OF NETWORK TRAFFIC

FIG. 2 is a block diagram of a networked computer system in which a method of  
20 selectively forced redirection of network traffic may be carried out.

As in the system of FIG. 1, a client 102 communicates with a network 104. In the preferred embodiment, client 102 executes a Web browser 103 and communicates with network 104 using HTTP requests and responses. A first server 208 and a second server 206 are coupled to network 104 and provide one or more network resources that can be accessed  
25 by the browser 103 using appropriate HTTP requests. For clarity, FIG. 2 does not show all possible logical data communication paths that may be established between client 102 and

servers 206, 208. Path 209 is a logical communication path that may be used when server 206 and server 208 communicate with one another.

Server 206 is coupled to a Server Proxy 210, which may be one or more software or hardware elements or processes contained within or communicating with server 206. In one embodiment, Server Proxy 210 is coupled to and interacts with a stream tokenizer 212 and an attribute list 214, however, these elements are not required.

Server 208 is coupled to or includes at least one resource 216 that is used by client 102. The resource 216 may be, for example, a Web page or a Web application, such as an interactive electronic commerce application. Similarly, server 206 is coupled to a resource 218, which may be a Web application or similar program.

In this configuration, Server Proxy 210 is configured to interact with server 206 to selectively force redirection of network traffic directed to server 208 from that server to server 206. In one embodiment, Server Proxy 210 is one or more software elements that are installed on and executed by server 206. In a preferred embodiment, Server Proxy 210 is configured with computer program instructions that cause the system of FIG. 2 to operate generally as follows.

Client 102 submits a request for resource 218 to the network 104. The request may be submitted in the form of a URL that includes the name of server 206, which hosts resource 218. The term "host" means to store or execute a resource, or manage execution of a resource located elsewhere.

Through means that are well known in this field, such as by using the DNS system, network 104 determines the location of server 206 and routes the client's request to that server. Server 206 receives and processes all requests by always redirecting them to server 208 over logical path 209. The redirection may be done selectively for particular types of requests or may be done for all requests received by server 206.

Server 208 then receives and processes the request. The request may require server 208 to invoke resource 216. In one embodiment, resource 216 executes and generates one or more HTML pages as a result. For example, when resource 216 is an electronic commerce application, it may generate a catalog page, or a confirmation page showing a customer order  
5 that is formatted in HTML for interpretation and presentation by browser 103. The HTML result may contain hyperlinks or other references to other Web pages, applications or resources hosted by server 208. Server 208 sends the HTML result to server 206.

Server Proxy 210 intercepts the HTML result. Using stream tokenizer 212, Server Proxy 210 scans the HTML result to identify references to resources hosted by server 208. If  
10 any such references are found, they are modified to refer to server 206 or to Server Proxy 210 and stored as a modified result. Server 206 sends the modified result back to client 102.

Accordingly, when a user of client 102 clicks on a hyperlink or other reference of the modified result, the hyperlink will always direct browser 103 to server 206 or to Server Proxy 210. In this way, server 206 retains control over the overall transaction with browser 103 and  
15 prevents the browser or client from interacting directly with server 208. As an example, Server Proxy 210 may be used to add Secure Sockets Layer (SSL) functions to services from a non-SSL Web server.

#### -- SERVER PROXY

In the preferred embodiment, Server Proxy 210 is implemented in the form of a  
20 program for use with a CGI script, which may be called ServerProxy.cgi, that is stored in association with server 206.

In general, redirection is accomplished by taking the path portion of a request sent to this program and passing the request onto the server which is being proxied for. For example, if the Server Proxy 210 is invoked by browser 103 using a URL of the form  
25 “http://aladdin/CScgi/ServerProxy.cgi/foo?bar=loo” and if aladdin:9090 is the server that is

being proxied, then Server Proxy 210 will translate the request into a URL of the form  
"http://aladdin:9090/foo?bar=loo."

Server Proxy 210 uses this URL to invoke Server 208 (aladdin:9090). After processing by Server 208, the result of the request is sent to Server Proxy 210. If the result is non-HTML, the result will be sent on to client 102 without modification. If the result is HTML, all URLs in the HTML that point to the proxied server are modified to point to the Server Proxy 210 instead. The modified result is then sent on to client 102.

For example, the HTML tag

<A HREF="http://aladdin:9090/foo?bar=loo">

would be translated to

<A HREF="http://aladdin/CScgi/ServerProxy.cgi/foo?bar=loo">

This forces any subsequent requests from the client based on the modified HTML to come back to the Server Proxy 210. Both HTTP methods PUT and GET are supported.

In one specific embodiment, Server Proxy 210 is implemented as a CGI script coupled to a computer program in the Java language, both integrated with the CiscoSecure access control server product, which is commercially available from Cisco Systems, Inc. The Server Proxy 210 functions to add SSL capability to a Web server that lacks such capability.

In this specific embodiment, the logic described above is followed unless a request to log on to a CiscoSecure server is made, for example, using a URL of the format

"/CScgi/ServerProxy.cgi/Logon". Special processing is used in this case, for performance enhancement. If SSL is not being used, the resulting HTML page is modified such that subsequent communication based on its URLs will occur directly between the proxied server and the client. The purpose for this is to optimize performance by taking Server Proxy 210 out of the link when SSL is not needed. Also, a special parameter required by the Logon service, 'SSL' (which may have the value 'true' or 'false') is appended to the request made to the proxied server.

A CGI script that invokes Server Proxy 210 is created and installed on the server that acts as proxy, for example, server 206 of FIG. 2. The CGI script is created and installed according to the requirements of that Web server. In addition to the regular CGI environment variables passed by the script, an 'actualServer' variable is passed, to tell Server Proxy 210 which server it is acting as a proxy for. A sample script is set forth in Table 1.

TABLE 1 -- EXAMPLE SCRIPT THAT INVOKES SERVER PROXY

```

10  #!/bin/sh
    BASE=/export/home/k2
    JBASE=/export/home/k2/java
    JAVA_HOME=$BASE/java
    JRE_TOP=$JAVA_HOME

15  JRE_CLASSES=$JRE_TOP/lib/rt.jar:$JRE_TOP/lib/i18n.jar:$JRE_TOP/lib/class
    es.jar:$JRE_TOP/lib/classes.zip:$JRE_TOP/classes

    CLASSPATH=/export/home/dstill/classes:$CLASSPATH:/export/home/k2/DBSer
    ver/DBServer.zip:/export/home/k2/FastAdmin/Acme.zip:/export/home/k2/FastAd
20  min/fastadmin.zip:$WEBLOGIC/classes:$JRE_CLASSES
    PATH=$PATH:$JAVA_HOME/bin
    export JAVA_HOME
    export JRE_TOP
    export JRE_CLASSES
25  export CLASSPATH
    export PATH
    cd $BASE/ns-home/docs/CScgi
    $JBASE/bin/java \
    -Dcgi.content_type=$CONTENT_TYPE \
30  -Dcgi.content_length=$CONTENT_LENGTH \
    -Dcgi.request_method=$REQUEST_METHOD \
    -Dcgi.query_string=$QUERY_STRING \
    -Dcgi.server_name=$SERVER_NAME \
    -Dcgi.server_port=$SERVER_PORT \
35  -Dcgi.script_name=$SCRIPT_NAME \
    -Dcgi.path_info=$PATH_INFO \
    -DactualServer="aladdin:9095" \
    cisco.ciscosecure.fastadmin.cgi.ServerProxy

```



-- STREAM TOKENIZER

Stream tokenizer 212 is a program or process that receives a text file as input, such as an HTML file, and produces a stream of tokens that represent tags and attributes within the contents of the input file. The tokens are produced according to token selection criteria that are coded into the stream tokenizer 212. An example of a suitable program that can be used for stream tokenizer 212 is the method java.io.StreamTokenizer, which is part of the Java Development Kit version 1.1.5, commercially available from Sun Microsystems, Inc. In off-the-shelf form, this Stream Tokenizer method is hard coded to treat character “\” in strings as an escape character. In HTML, however, the character “\” is not defined as an escape character and needs to be part of appropriate tokens. Accordingly, in the preferred embodiment, the method java.io.StreamTokenizer of the JDK is modified to enable such characters to appear in tokens. This may be done by commenting out the code that handles escape functions.

-- METHOD OF SELECTIVELY FORCING REDIRECTION OF NETWORK TRAFFIC

FIG. 3A is a flow diagram of one embodiment of a method of selectively forcing redirection of network traffic. For convenience, the method of FIG. 3A is described below in connection with the structural elements of FIG. 2, however, the method of FIG. 3A does not require use of such a structure.

In block 302, a client request is received. For example, client 102 submits a request for resource 218 to the network 104 in the form of a URL that includes the name of server 206. Network 104 determines the location of server 206 and routes the client’s request to that server. In block 304, the client request is automatically redirected to another server that actually processes the request and generates a response. For example, server 206 redirects the client request it to server 208 over logical path 209.

In block 306, the request is processed and a response is generated. For example, server 208 then receives and processes the request, in part by invoking resource 216, which generates one or more HTML pages as a result. The HTML result may contain hyperlinks or other references to other Web pages, applications or resources hosted by server 208. Server  
5 208 sends the HTML result to server 206. As indicated in block 308, the response is received, for example, at Server Proxy 210.

In block 310, the process determines whether the response includes HTML results. HTML is merely an example of a format or protocol in which results may be presented. The method is equally applicable to any type of results that may internally reference a resource  
10 that is associated with the thing that generates the results.

If the response does contain HTML results, then control is passed to block 312 in which the process translates references within the HTML results. For example, when the results are generated by server 208 and reference resources associated with that server, block 312 may involve the steps of translating the references into modified references that identify  
15 server 206, Server Proxy 210, or the process of FIG. 3A itself. In one embodiment, using stream tokenizer 212, Server Proxy 210 scans the HTML result to identify references to resources hosted by server 208. If any such references are found, they are modified to refer to server 206 or to Server Proxy 210 and stored as a modified result.

In block 314, server 206 sends the modified result back to client 102.

20 FIG. 3B is a flow diagram of an embodiment of steps that may be carried out at block 312 of FIG. 3A.

In block 320, a result file that contains the results is parsed into one or more tokens. Block 320 may involve, for example, opening an input result file, opening an output result file, and using stream tokenizer 212 to scan the results to identify tokens in the input result  
25 file. As each token is identified, the steps of block 322 to block 332 are performed. Each token may comprise a tag and zero or more attributes associated with the tag.

10 For example, when a token is associated with a URL that is expressed in the HTML relative URL form, meaning that it refers to a directory of the server that created the results, then that token is identified as a candidate for translation. Also, if a token contains an express reference to that server, such as a URL that literally contains the server's name, then that token is also identified as a candidate for translation.

15           Tokens that are not candidates for translation are written or otherwise output into the  
output result file, as shown in block 330.

When a token is identified as a candidate for translation, then control is passed to block 328, in which a reference to a script is prepended to the URL attribute of the token. For example, block 328 may involve prepending the name of a CGI script, which implements the process of FIG. 3A and FIG. 3B, to the URL attribute of the token. Control is then passed to block 330 in which the token is output.

Subsequently, when a user of client 102 clicks on a hyperlink or other reference of the modified result, the hyperlink will always direct browser 103 to server 206 or to Server Proxy 210. Server 206 will then redirect the browser 103 to server 208, which processes the subsequent request. If HTML is generated again, then Server Proxy 210 will translate any references to server 208 into references to server 206. In this way, server 206 retains control

over the overall transaction with browser 103 and prevents the browser or client from interacting directly with server 208.

#### -- HARDWARE OVERVIEW

FIG. 4 is a block diagram that illustrates a computer system 400 upon which an embodiment of the invention may be implemented. Computer system 400 includes a bus 402 or other communication mechanism for communicating information, and a processor 404 coupled with bus 402 for processing information. Computer system 400 also includes a main memory 406, such as a random access memory (RAM) or other dynamic storage device, coupled to bus 402 for storing information and instructions to be executed by processor 404. Main memory 406 also may be used for storing temporary variables or other intermediate information during execution of instructions to be executed by processor 404. Computer system 400 further includes a read only memory (ROM) 408 or other static storage device coupled to bus 402 for storing static information and instructions for processor 404. A storage device 410, such as a magnetic disk or optical disk, is provided and coupled to bus 402 for storing information and instructions.

Computer system 400 may be coupled via bus 402 to a display 412, such as a cathode ray tube (CRT), for displaying information to a computer user. An input device 414, including alphanumeric and other keys, is coupled to bus 402 for communicating information and command selections to processor 404. Another type of user input device is cursor control 416, such as a mouse, a trackball, or cursor direction keys for communicating direction information and command selections to processor 404 and for controlling cursor movement on display 412. This input device typically has two degrees of freedom in two axes, a first axis (e.g., x) and a second axis (e.g., y), that allows the device to specify positions in a plane.

The invention is related to the use of computer system 400 for selectively forced redirection of network traffic. According to one embodiment of the invention, selectively

forced redirection of network traffic is provided by computer system 400 in response to processor 404 executing one or more sequences of one or more instructions contained in main memory 406. Such instructions may be read into main memory 406 from another computer-readable medium, such as storage device 410. Execution of the sequences of  
5 instructions contained in main memory 406 causes processor 404 to perform the process steps described herein. In alternative embodiments, hard-wired circuitry may be used in place of or in combination with software instructions to implement the invention. Thus, embodiments of the invention are not limited to any specific combination of hardware circuitry and software.

The term "computer-readable medium" as used herein refers to any medium that  
10 participates in providing instructions to processor 404 for execution. Such a medium may take many forms, including but not limited to, non-volatile media, volatile media, and transmission media. Non-volatile media includes, for example, optical or magnetic disks, such as storage device 410. Volatile media includes dynamic memory, such as main memory 406. Transmission media includes coaxial cables, copper wire and fiber optics, including the  
15 wires that comprise bus 402. Transmission media can also take the form of acoustic or light waves, such as those generated during radio-wave and infra-red data communications.

Common forms of computer-readable media include, for example, a floppy disk, a flexible disk, hard disk, magnetic tape, or any other magnetic medium, a CD-ROM, any other optical medium, punchcards, papertape, any other physical medium with patterns of holes, a  
20 RAM, a PROM, and EPROM, a FLASH-EPROM, any other memory chip or cartridge, a carrier wave as described hereinafter, or any other medium from which a computer can read.

Various forms of computer readable media may be involved in carrying one or more sequences of one or more instructions to processor 404 for execution. For example, the instructions may initially be carried on a magnetic disk of a remote computer. The remote  
25 computer can load the instructions into its dynamic memory and send the instructions over a telephone line using a modem. A modem local to computer system 400 can receive the data

on the telephone line and use an infrared transmitter to convert the data to an infrared signal. An infrared detector can receive the data carried in the infrared signal and appropriate circuitry can place the data on bus 402. Bus 402 carries the data to main memory 406, from which processor 404 retrieves and executes the instructions. The instructions received by  
5 main memory 406 may optionally be stored on storage device 410 either before or after execution by processor 404.

Computer system 400 also includes a communication interface 418 coupled to bus 402. Communication interface 418 provides a two-way data communication coupling to a network link 420 that is connected to a local network 422. For example, communication  
10 interface 418 may be an integrated services digital network (ISDN) card or a modem to provide a data communication connection to a corresponding type of telephone line. As another example, communication interface 418 may be a local area network (LAN) card to provide a data communication connection to a compatible LAN. Wireless links may also be implemented. In any such implementation, communication interface 418 sends and receives  
15 electrical, electromagnetic or optical signals that carry digital data streams representing various types of information.

Network link 420 typically provides data communication through one or more networks to other data devices. For example, network link 420 may provide a connection through local network 422 to a host computer 424 or to data equipment operated by an  
20 Internet Service Provider (ISP) 426. ISP 426 in turn provides data communication services through the worldwide packet data communication network now commonly referred to as the "Internet" 428. Local network 422 and Internet 428 both use electrical, electromagnetic or optical signals that carry digital data streams. The signals through the various networks and the signals on network link 420 and through communication interface 418, which carry the  
25 digital data to and from computer system 400, are exemplary forms of carrier waves transporting the information.

Computer system 400 can send messages and receive data, including program code, through the network(s), network link 420 and communication interface 418. In the Internet example, a server 430 might transmit a requested code for an application program through Internet 428, ISP 426, local network 422 and communication interface 418. In accordance  
5 with the invention, one such downloaded application provides for selectively forced redirection of network traffic as described herein.

The received code may be executed by processor 404 as it is received, and/or stored in storage device 410, or other non-volatile storage for later execution. In this manner, computer system 400 may obtain application code in the form of a carrier wave.

10 -- SCOPE

In the foregoing specification, the invention has been described with reference to specific embodiments thereof. It will, however, be evident that various modifications and changes may be made thereto without departing from the broader spirit and scope of the invention. For example, the invention and claims do not require use of HTML, HTTP, or  
15 TCP/IP. These are merely convenient examples of formats and protocols that may be used. The invention encompasses and is fully applicable to, for example, results that are expressed in any hypertext format, or any other format that may contain references to the thing that created it.

The specification and drawings are, accordingly, to be regarded in an illustrative  
20 rather than a restrictive sense.

---

## CLAIMS

What is claimed is:

- 1 1. A method of securely communicating information in a network that includes a host  
2 that originates a request, a first server that serves a response to the request, and a  
3 second server that cooperates with the first server to respond to the request, the  
4 method comprising the computer-implemented steps of:  
5 receiving a first request for a service from the host, which request includes a network  
6 address of the host; and  
7 communicating a second service request to the second server based on the first service  
8 request and including the host network address only when a first network  
9 address of the first server is identical to a second network address of the  
10 second server.
- 1 2. A method as recited in Claim 1, wherein the request of the host comprises a key value  
2 comprising an originating host Internet Protocol (IP) address and a random value.
- 1 3. A method as recited in Claim 1, wherein the step of communicating a second service  
2 request comprises the step of accepting the host request only when an IP address of  
3 the second server is the same as an IP address of the first server.
- 1 4. A method as recited in Claim 1, wherein the host is a Web browser and wherein the  
2 host request comprises a Universal Resource Locator (URL) that includes an IP  
3 address of the host.
- 1 5. A method as recited in Claim 1, wherein the host is a Web browser and wherein the  
2 host request comprises an HTML POST form that includes an IP address of the host.
- 1 6. A method of securely communicating data between a proxy server and a second  
2 server, wherein each of the proxy server and the second server are addressable by first



3 and second Internet Protocol (IP) addresses, respectively, the method comprising the  
4 computer-implemented steps of:  
5 receiving, at the proxy server, a first service request from a browser client, wherein the  
6 service request includes a third IP address of a client computer associated with  
7 the browser client;  
8 communicating a second service request that includes the browser client IP address to  
9 the second server only when the first IP address of the proxy server is identical  
10 to the second IP address of the second server.

1 7. A method as recited in Claim 6, wherein the first service request of the browser client  
2 comprises a key value comprising the third IP address and a random value.

1 8. A method as recited in Claim 6, wherein the first service request of the browser client  
2 comprises a Universal Resource Locator (URL) that includes an IP address of the  
3 host.

1 9. A method as recited in Claim 6, wherein the first service request of the browser client  
2 comprises an HTML POST form that includes an IP address of the host.

1 10. A method of securely communicating a network address of a client that issues service  
2 requests to a first server that proxies the service requests for a second server,  
3 comprising the computer-implemented steps of:  
4 receiving a network address of the client;  
5 determining whether a first network address of the first server is equal to a second  
6 network address of the second server; and  
7 sending the network address of the client from the first server to the second server in a  
8 secure request message only when the first network address of the first server  
9 is equal to the second network address of the second server.

- 1 11. A method as recited in Claim 10, wherein each of the service requests of the browser  
2 client comprises a key value comprising an IP address of the client and a random  
3 value.
- 1 12. A method as recited in Claim 10, wherein each of the service requests of the browser  
2 client comprises a Universal Resource Locator (URL) that includes an IP address of  
3 the browser client.
- 1 13. A method as recited in Claim 10, wherein each of the service requests of the browser  
2 client comprises an HTML POST form that includes an IP address of the browser  
3 client.
- 1 14. A data communications apparatus that securely communicates a service request that is  
2 received from a client, comprising a first server that proxies the service request for a  
3 second server, the first server comprising means for receiving a network address of the  
4 client; means for determining whether a first network address of the first server is  
5 equal to a second network address of the second server; and means for sending the  
6 network address of the client from the first server to the second server in a secure  
7 request message only when the first network address of the first server is equal to the  
8 second network address of the second server.
- 1 15. An apparatus as recited in Claim 14, wherein the service request comprises a key  
2 value comprising an IP address of the client and a random value.
- 1 16. An apparatus as recited in Claim 14, wherein the service request comprises a  
2 Universal Resource Locator (URL) that includes an IP address of the browser client.

- 1 17. An apparatus as recited in Claim 14, wherein the service request comprises an HTML  
2 POST form that includes an IP address of the browser client.
- 1 18. A computer-readable medium carrying one or more sequences of instructions for  
2 securely communicating a network address of a client that issues service requests to a  
3 first server that proxies the service requests for a second server, wherein execution of  
4 the one or more sequences of instructions by one or more processors causes the one or  
5 more processors to perform the steps of:  
6 receiving a network address of the client;  
7 determining whether a first network address of the first server is equal to a second  
8 network address of the second server; and  
9 sending the network address of the client from the first server to the second server in a  
10 secure request message only when the first network address of the first server  
11 is equal to the second network address of the second server.
- 1 19. A computer-readable medium as recited in Claim 18, wherein each of the service  
2 requests of the browser client comprises a key value comprising an IP address of the  
3 client and a random value.
- 1 20. A computer-readable medium as recited in Claim 18, wherein each of the service  
2 requests of the browser client comprises a Universal Resource Locator (URL) that  
3 includes an IP address of the browser client.
- 1 21. A computer-readable medium as recited in Claim 18, wherein each of the service  
2 requests of the browser client comprises an HTML POST form that includes an IP  
3 address of the browser client.

- 1 22. A data communications apparatus that securely communicates a service request that is  
2 received from a client, comprising:  
3 a first server that proxies the service request for a second server comprising a network  
4 interface to a network that includes the first server and the second server;  
5 a processor in the first server;  
6 a storage device in the first server comprising one or more sequences of stored  
7 instructions which, when executed by the processor, cause the processor to  
8 carry out the steps of:  
9 receiving a network address of the client;  
10 determining whether a first network address of the first server is equal to a  
11 second network address of the second server; and  
12 sending the network address of the client from the first server to the second  
13 server in a secure request message only when the first network address of the  
14 first server is equal to the second network address of the second server.
- 1 23. An apparatus as recited in Claim 22, wherein the service request comprises a key  
2 value comprising an IP address of the client and a random value.
- 1 24. An apparatus as recited in Claim 22, wherein the service request comprises a  
2 Universal Resource Locator (URL) that includes an IP address of the browser client.
- 1 25. An apparatus as recited in Claim 22, wherein the service request comprises an HTML  
2 POST form that includes an IP address of the browser client.

## ABSTRACT OF THE DISCLOSURE

A method of securely communicating a network address of a client that issues service requests to a first server that proxies the service requests for a second server. A network address of the client is received. A processor determines whether a first network address of the first server is equal to a second network address of the second server. The network address of the client is sent from the first server to the second server in a secure request message only when the first network address of the first server is equal to the second network address of the second server. Accordingly, a secure communications protocol is provided in which an address of a requesting client, e.g., an IP address, is passed in the protocol only among a responding server and its proxy, thereby preventing interception of the client IP address by unauthorized processes. By enforcing a policy that permits the network address of an originating host to pass from a first server to a second server only when the network address of the second server meets specified criteria (e.g., it is the same network address as that of the first server), the originating host address can be passed securely through a proxy server.

FIG. 1A

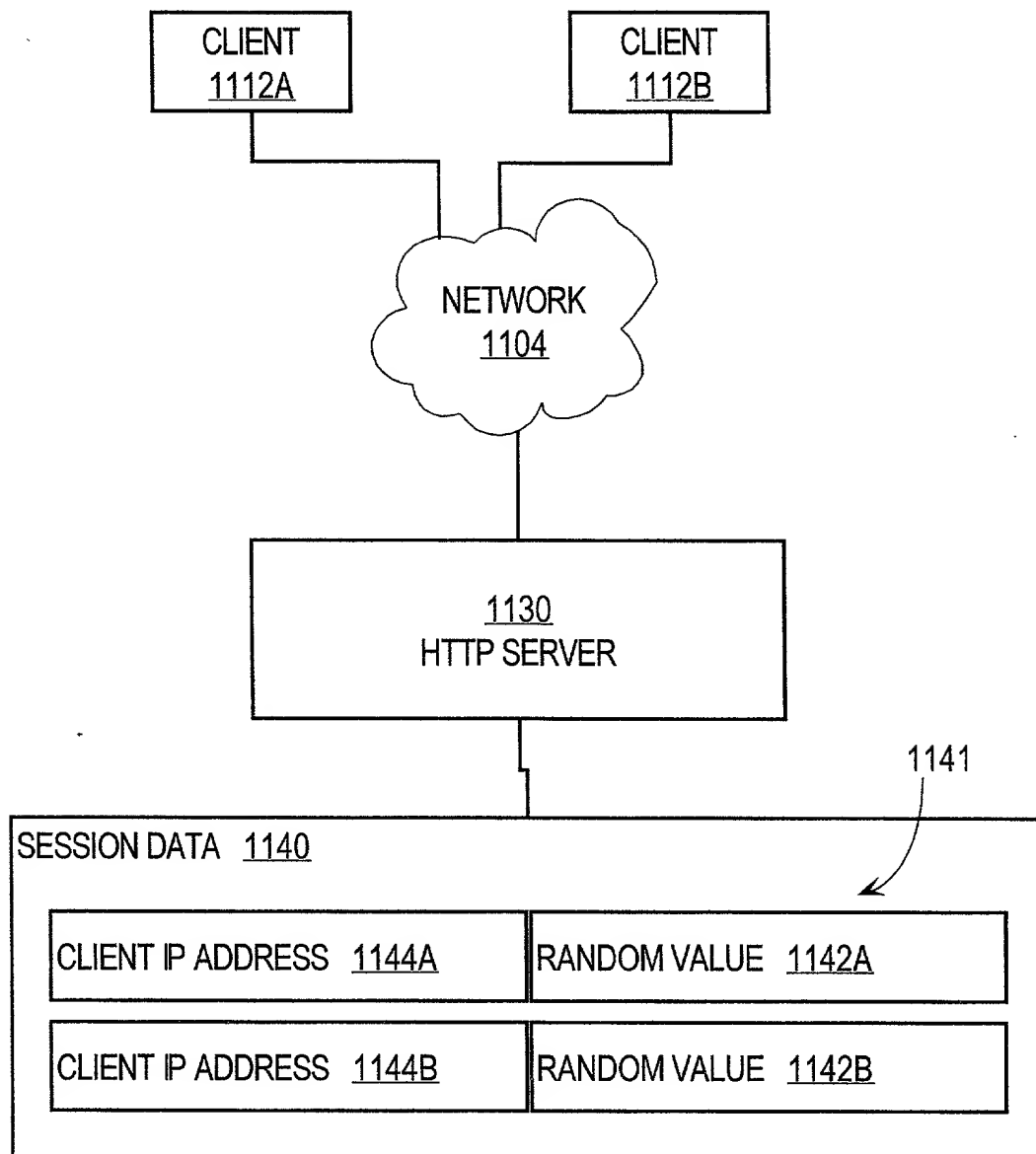
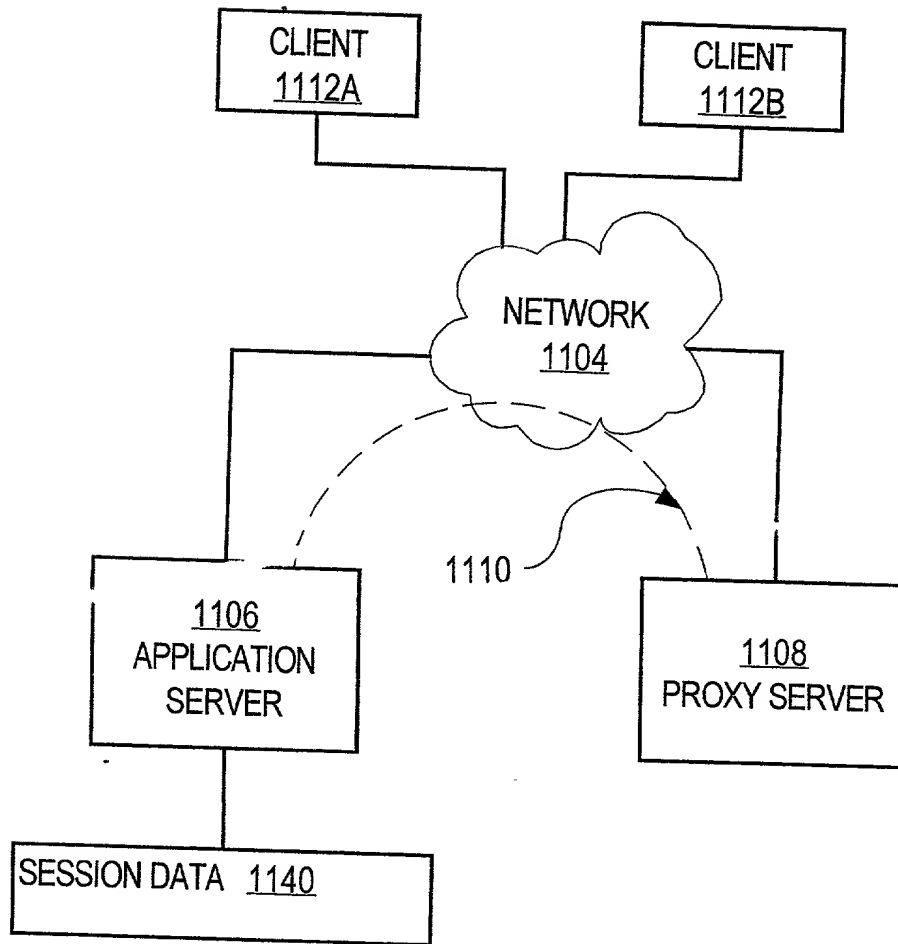


FIG. 1B



**FIG. 1C (PRIOR ART)**

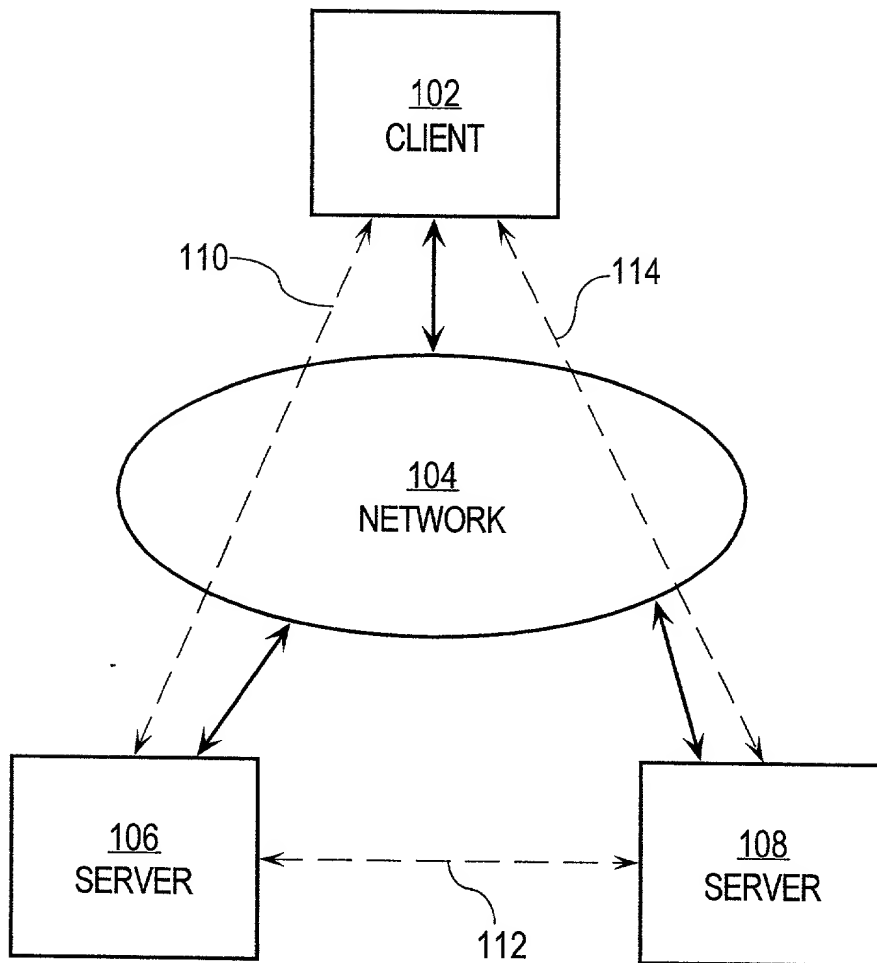




FIG. 2

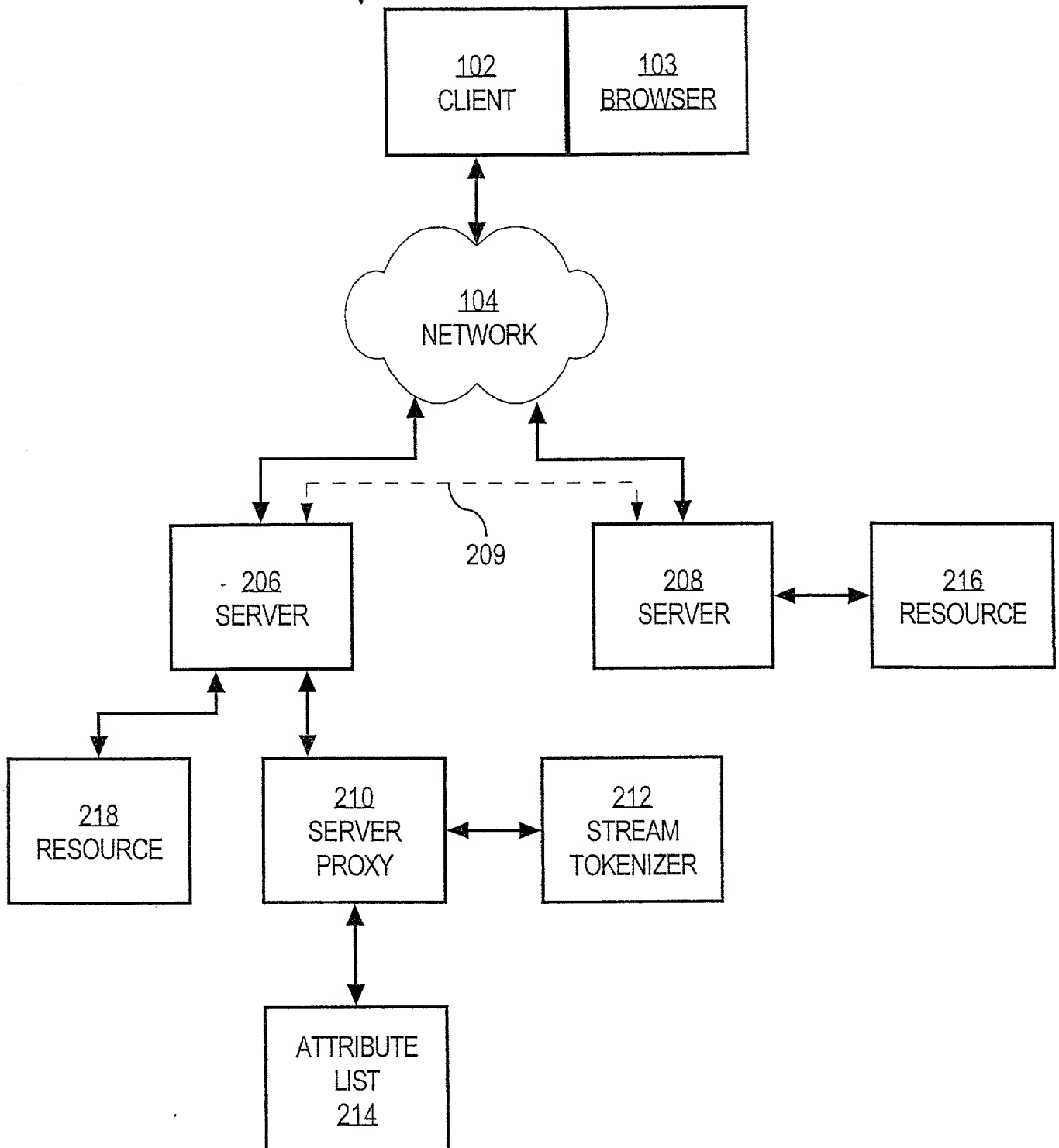


FIG. 3A

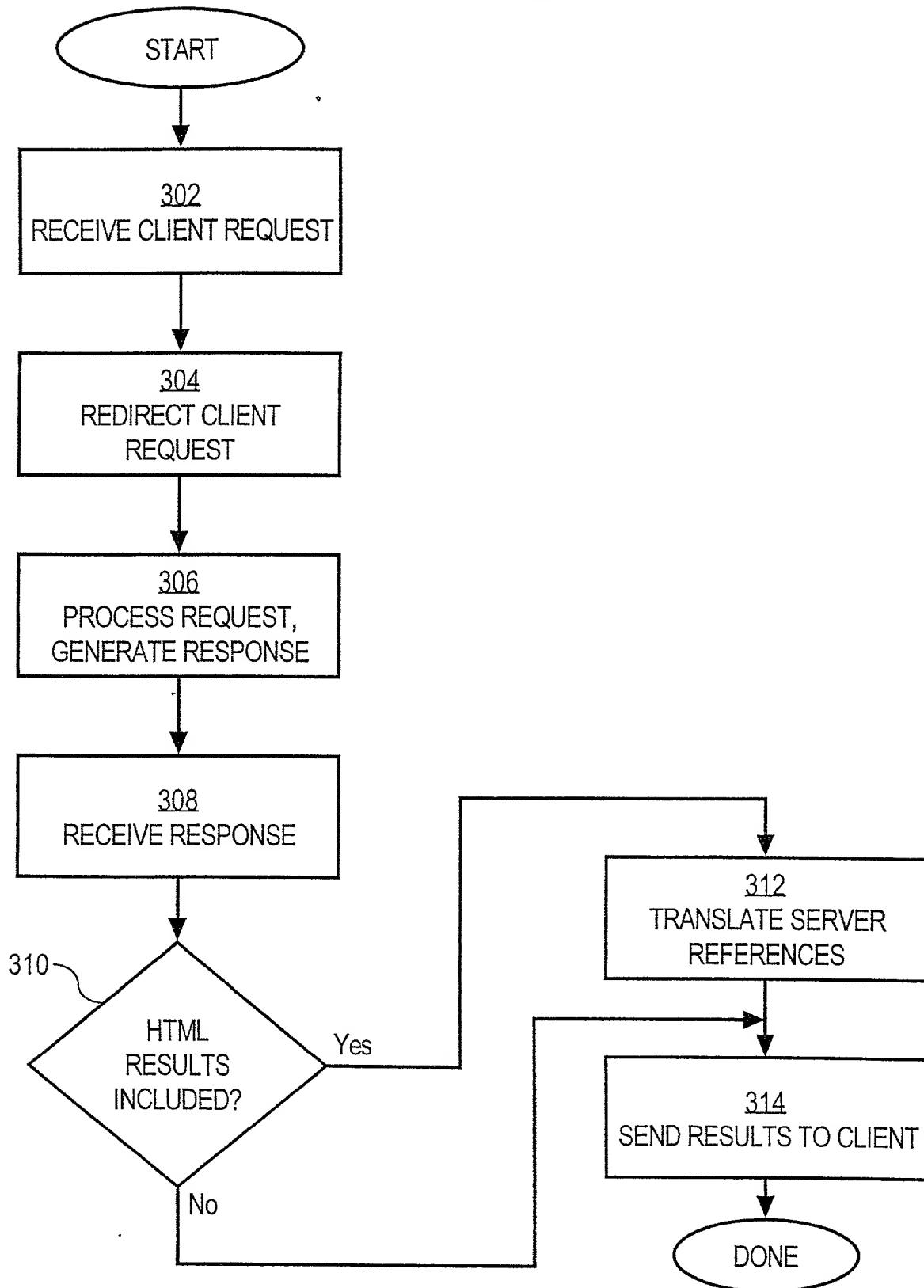


FIG. 3B

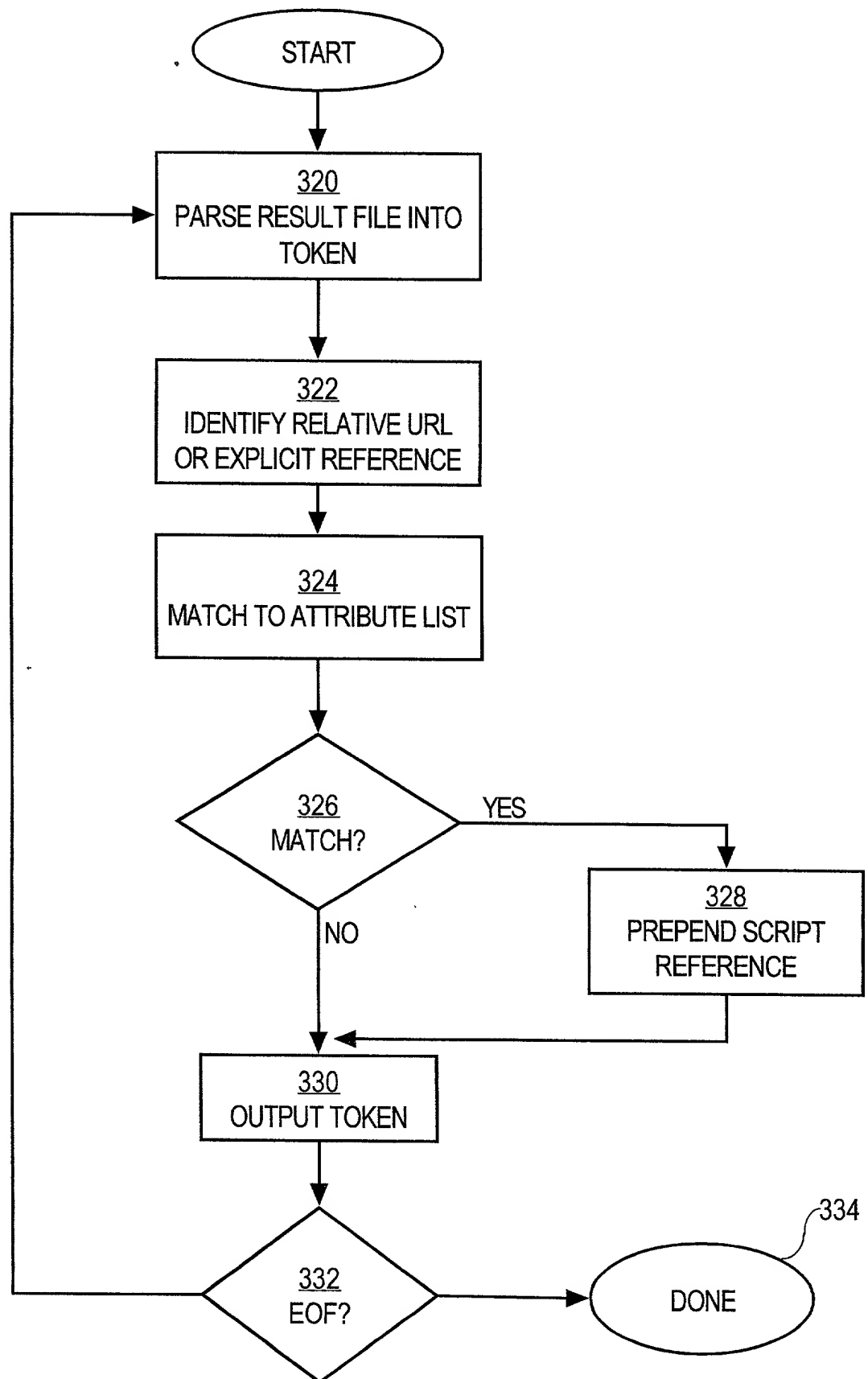
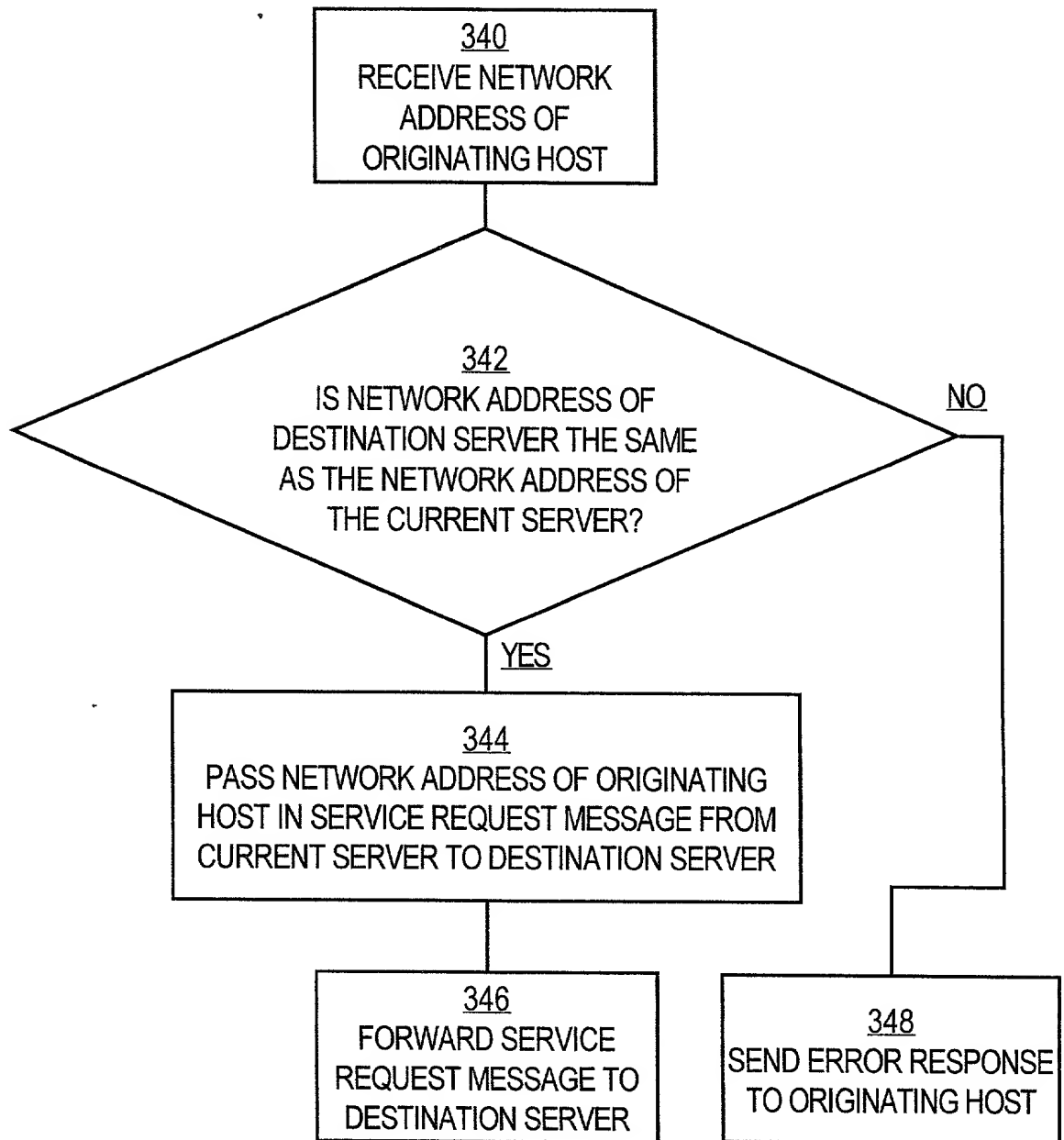
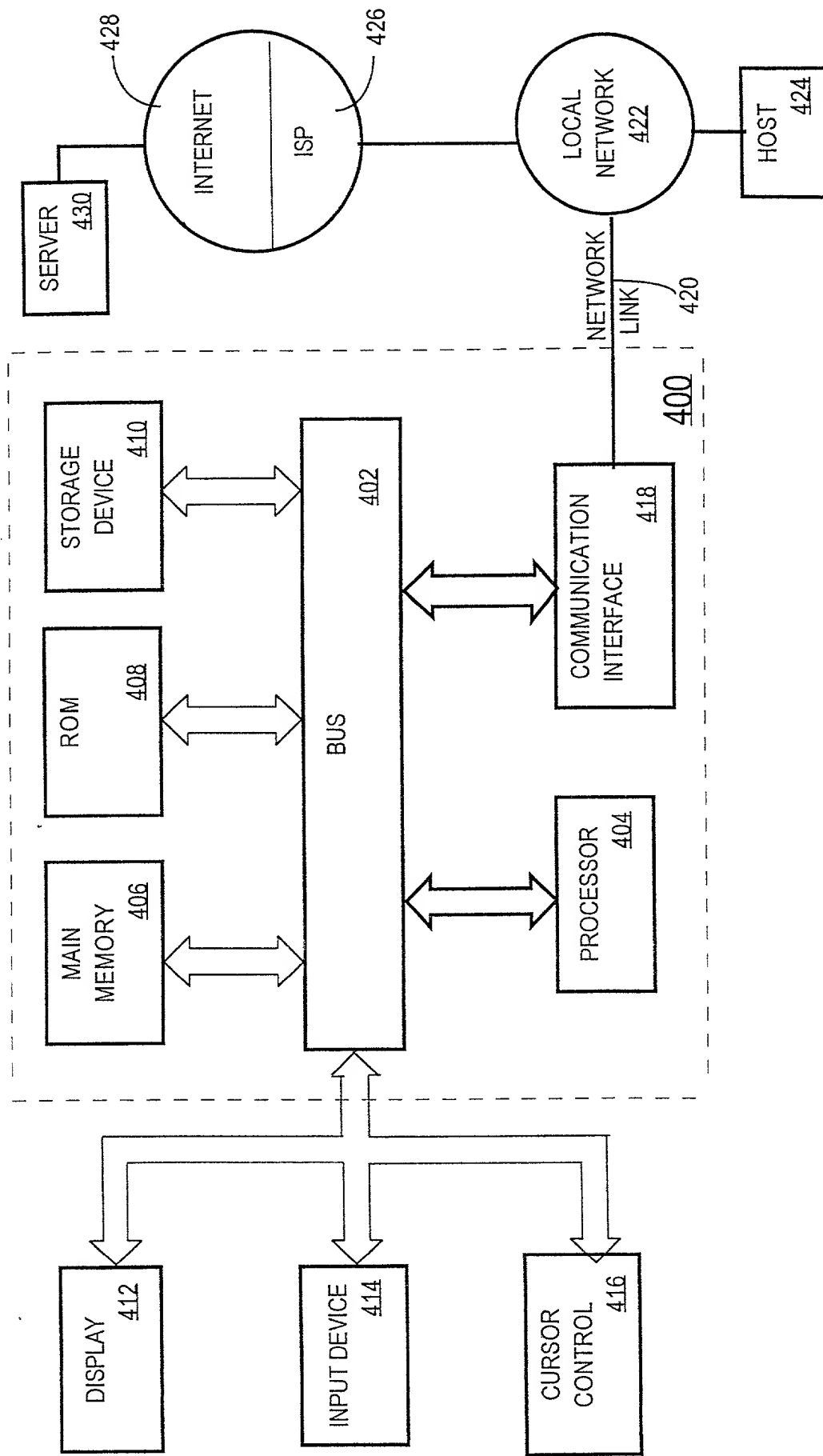


FIG. 3C



006080" 26E9E960

FIG.4



**DECLARATION AND POWER OF ATTORNEY**

As a below named inventor, I hereby declare that:

My residence, post office and citizenship are as stated below next to my name,

I believe I am an original, first and joint inventor of the subject matter claimed and for which a patent is sought on the invention SELECTIVELY PASSING NETWORK ADDRESSES THROUGH A SERVER, the specification of which

☐ is attached hereto.

☒ was filed on \_\_\_\_\_ as Application Serial No. \_\_\_\_\_ and was amended on (if applicable).

I hereby state that I have reviewed and understand the contents of the above identified specification, including the claims, as amended by any amendment referred to above.

I acknowledge the duty to disclose information which is known to me to be material to patentability in accordance with Title 37, Code of Federal Regulations, Section 1.56.

I hereby claim foreign priority benefits under Title 35, United States Code, Section 119 of any foreign application(s) for patent or inventor's certificate listed below and have also identified below any foreign application for patent or inventor's certificate having a filing date before that of the application on which priority is claimed:

**Prior Foreign Applications(s):**

Number	Country	Day/Month/Year filed	Priority Claimed
			<input type="checkbox"/>
			<input type="checkbox"/>

I hereby claim the benefit under 35 USC §119(e) of any United States provisional application(s) listed below.

**Prior Provisional Application(s):**

Application Number	Filing Date
--------------------	-------------

I hereby claim the benefit under Title 35, United States Code, Section 120 of any United States application(s) listed below and, insofar as the subject matter of each of the claims of this application is not disclosed in the prior United States application in the manner provided by the first paragraph of Title 35, United States Code, Section 112, I acknowledge the duty to disclose material information as defined in Title 37, Code of Federal Regulations, Section 1.56 which occurred between the filing date of the prior application and the national or PCT international filing date of this application:



Docket No.: 50325-0114

---

Full name of third inventor: THOMAS A. RODEN

---

Inventor's signature:

Date:

---

Residence: 16 Villanova, Irvine, California 92606-8929

---

Citizenship: USA

---

Post Office Address:

---

006080" 26E9E9E0